IITM-CS6840: Advanced Complexity Theory January 28, 2012

Lecture 12 : One random string for all

Lecturer: Jayalal Sarma M.N. *Scribe: Sajin Koroth*

THEME: Advice Classes and BPP

LECTURE PLAN:Today we will be showing an interesting consequence of amplification of BPP introduced earlier. We will show that for an amplified BPP algorithm there is a good string of random bits for each input length n such that the algorithm run with these random bits is correct for all inputs x of length n. Hence if you could get this good random string some how then you can decide a language in BPP in polynomial time without any randomness. But there is a catch, although we prove the existence of such a random string we do not know how to compute such a string efficiently. Hence we will introduce a new model of computation where you are given such advice strings for free, but the advice for all inputs of length n has to be the same. We will introduce an advice string based class called $P/poly$, and will discuss its connection to BPP

1 One random string for all

Recall that amplification allows to transform in polynomial time any BPP algorithm to a BPP algorithm with error bound 2^{-2n}(i.e. at most 2^{-2n} fraction of random strings are "bad") using $p(n)$ randomness. We will show that for a BPP algorithm with the above mentioned error bound there is one random string for every length n such that for any input of that length n, the BPP algorithm outputs correctly on that random string. For the rest of the lecture we will work with sufficiently amplified success probability BPP machines, where the notion of sufficient success probability is defined as given below :

$$x \in L \Rightarrow \Pr_y [M(x,y) \text{ accepts }] \geq 1 - 2^{-2n} \tag{1}$$

$$x \notin L \Rightarrow \Pr_y [M(x,y) \text{ accepts }] \geq 2^{-2n} \tag{2}$$

That is in such a machine the number of random strings y which lead the machine to output a wrong answer is bounded by 2^{-2n}. Now let us consider a matrix A whose rows are indexed by inputs of length n and columns are indexed by random strings of length $p(n)$, and the (i,j)th entry is 1 if on fixing the random bits to be j the machine M on input i outputs correctly and it is 0 otherwise. That is $A(i,j) = 1$ if and only if $M(i,j) = \chi_L(i)$, where $\chi_L(i)$ is the membership function of the language L (i.e. $\chi_L(i) = 1$ if and only if $i \in L$). By the amplification we are guaranteed that for a given input i at most 2^{-2n} fraction of the random strings can have $A(i,j) = 0$. Hence the total number of zeros in the A matrix is at

most the number of rows times the maximum number of zeros in a row, which is equal to

$$\begin{aligned} \#\ 0\text{'s in matrix } A &\leq 2^n \times 2^{-2n} \times 2^{p(n)} \\ &\leq 2^{p(n)-n} \end{aligned}$$

But the total number of zeros, $2^{p(n)-n}$ is strictly less than the number of columns in the matrix A. Hence there must be at least one column with no zeros in it. If a column in the A matrix has no zeros then by the definition of A matrix, the random string represented by this column when fed as random bits to machine M would output correctly $\chi_L(x)$ for every $x \in \{0,1\}^n$.

2 Class P/poly

Even though we have proved the existence of a fixing of random bits for an arbitrary input length n of an amplified BPP machine M such that the M on these random bits decides all inputs x of a given length correctly for $L(M)$, we do not know how to compute such a string efficiently (deterministically or using a randomized algorithm) for arbitrary amplified BPP machines. Also note that the good random string can vary with the input length. But if we can get this random string for each input length n for **free** then we can decide a language in BPP in P. That if there is a function $h : N \to \{0,1\}^*$ such that $h(n)$ is at most polynomial in n and is the correct random string for the given BPP machine M, for all inputs of length n, for all n then we can construct a machine M' such that it on input $(x, h(|x|))$ will simulate M on x using $h(|x|)$ as the random bits tape.

We will generalize the above ideas to define a class such that every language in BPP is also in this class.

Definition 1 (P/poly)**.** A language L is in P/poly if there exists a polynomial $p(n)$, an advice function $h : N \to \{0,1\}^*$ and a language $B \in \mathsf{P}$ such that $\forall n, |h(n)| \leq p(n)$ and

$$x \in L \iff (x, h(|x|)) \in B$$

where $|x|$ denotes the length of the string x.

2.1 BPP $\subset$ P/poly

This is a straight forward corollary of the existence of a good random string for any BPP machine, which works correctly for all inputs of a given length. To show that for any $L \in$ BPP it is also true that $L \in$ P/poly we will use the fact that there a BPP machine M_L accepting L with error at most 2^{-2n} using at most $p(n)$ random bits. We have already shown that for such a machine for every input length n at least one of $2^{p(n)}$ possible random

strings is good for all inputs of length n. We define the advice function $h(n)$ to be a good random string which works for all inputs of length n. Hence $|h(n)| = p(n)$ is at most polynomial in input length. Note that definition of $P/poly$ doesn't have any requirements on the computability of such a function, but needs the guarantee that such a function exists. We will construct a machine M_B running in deterministic polynomial time which would accept the language $B \in \mathsf{P}$ which accepts $(x, h(|x|))$ for all $x \in L$. The machine M_B on input $(x, h(|x|))$ starts simulating M_L on input x using $h(|x|)$ as the random bits. By the definition of $h(|x|)$, $M_L(x)$ using random bits $h(|x|)$ accepts if and only if $x \in L$. Hence the proof.

2.2 With advice comes the undecidable

A consequence of the above definition of class $\mathsf{P/poly}$ is that it not only contains BPP, but it also contains some undecidable languages as we do not insist on computability of advice function h. One such undecidable language is **Unary Halting Problem** defined as

$$\text{UHP} = \{1^n \mid \text{ Turing machine encoded by bin}(n) \text{ halts on all inputs}\}$$

It is easy to note that the general halting problem reduces to the unary halting problem. Hence UHP is undecidable because HP is.

We can also show that UHP is in $\mathsf{P/poly}$. This is very straight forward because the language is a unary language and there is exactly one input of length n. Hence the advice function is simply a bit representing the answer to the UHP on input 1^n. Since we just need a single bit of advice note that UHP is also in $\mathsf{P}/\theta(1)$. Also from the above argument we can deduce that complement of UHP is also in $\mathsf{P/poly}$ because by modifying the $\mathsf{P/poly}$ machine for UHP to accept when $h(n) = 0$ and reject otherwise where $h()$ is the advice function for UHP, we get a $\mathsf{P/poly}$ machine for $\overline{\text{UHP}}$. Hence $\mathsf{P/poly}$ not only contains complete problems for semi-decidable languages, like UHP also contains languages which are complete for co-semi-decidable languages, like $\overline{\text{UHP}}$.